Hoofdstuk 8

Grafische afbeeldingen met een schildpad

Logo

In de jaren 70 was er een eenvoudige, maar krachtige programmeertaal met de naam Logo die werd gebruikt door een klein aantal onderzoekers. Totdat iemand 'Turtle Graphics' aan de taal toevoegde en een 'schildpad' op het scherm zichtbaar maakte die reageerde op opdrachten als *Ga vooruit, Ga rechtsaf, Ga linksaf.* enz. Met de schildpad konden mensen interessante vormen op het schermen tekenen. Dit maakte de taal onmiddellijk toegankelijk en aantrekkelijk voor mensen van alle leeftijden en de taal werd daarom razend populair in de jaren 80.

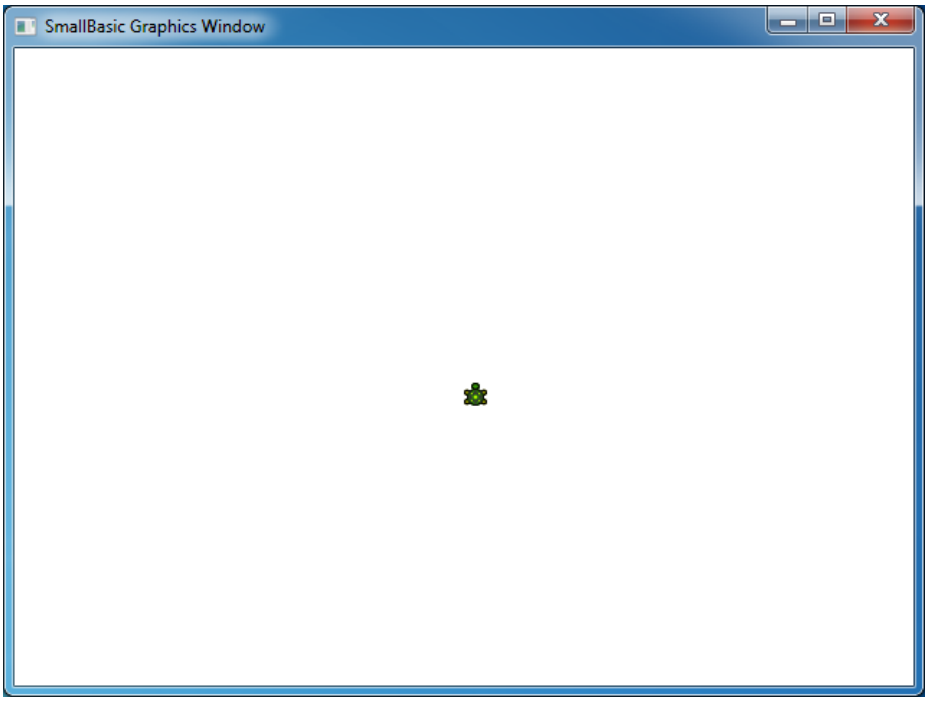
Small Basic heeft een **Turtle**-object met vele opdrachten die kunnen worden opgeroepen binnen Small Basic-programma's. In dit hoofdstuk zullen we de schildpad gebruiken voor het tekenen van grafische afbeeldingen op het scherm.

De schildpad

Voordat we kunnen beginnen, moeten we de schildpad zichtbaar maken op het scherm. Dit kunnen we doen met een eenvoudig eenregelig programma.

```
Turtle.Show()
```

Als je dit programma uitvoert, zul je een wit venster zien, net als het venster dat je in het vorige hoofdstuk zag, behalve dat er nu een schildpad in het midden van het venster is geplaatst. Deze schildpad gaat onze instructies volgen en tekenen wat we willen dat er wordt getekend.



Afbeelding 37 – Schildpad is zichtbaar

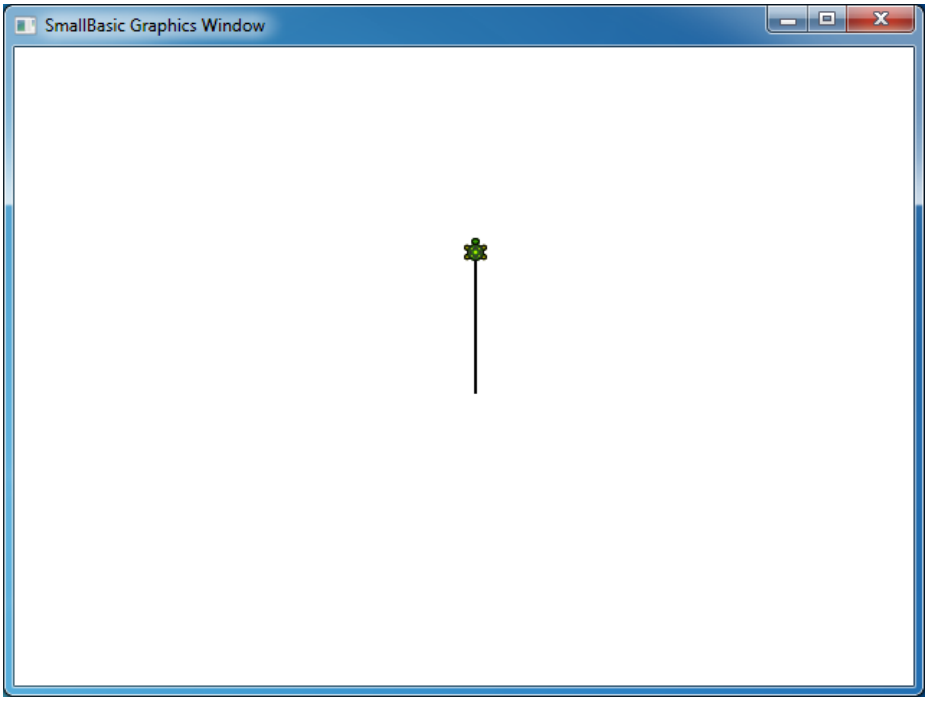
Verplaatsen en tekenen

Een van de instructies die je met de schildpad kunt uitvoeren is **Move**. Je moet een getal opgeven als de invoer voor deze bewerking. Dit getal geeft aan hoever de schildpad moet worden verplaatst. In het voorbeeld hieronder gaan we de schildpad 100 pixels verplaatsen.

```
Turtle.Move(100)
```

Als je dit programma uitvoert, kun je zien dat de schildpad langzaam 100 pixels naar boven wordt verplaatst. Tijdens deze verplaatsing zie je ook dat er een lijn wordt getekend achter de schildpad. Als je klaar bent met het verplaatsen van de schildpad, ziet het resultaat eruit als in de afbeelding hieronder.

Wanneer je bewerkingen van de schildpad gebruikt, hoef je Show() niet op te roepen. De schildpad wordt automatisch zichtbaar gemaakt wanneer er een Turtle-bewerking wordt uitgevoerd.



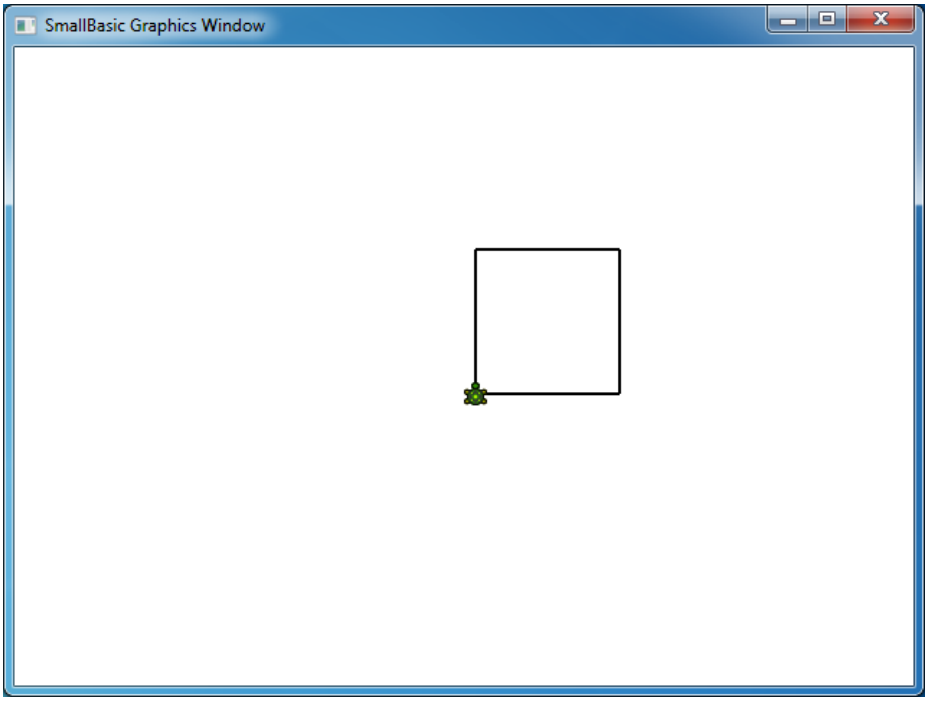
Afbeelding 38 – Honderd pixels verplaatsen

Een vierkant tekenen

Een vierkant heeft vier zijkanten, twee verticaal en twee horizontaal. Als we een vierkant willen tekenen, moeten we de schildpad een lijn laten tekenen, rechtsaf laten gaan, een andere lijn laten teken en rechtsaf laten gaan totdat alle vier zijkanten zijn getekend. In een programma ziet dit er als volgt uit:

```
Turtle.Move(100)
Turtle.TurnRight()
Turtle.Move(100)
Turtle.TurnRight()
Turtle.Move(100)
Turtle.TurnRight()
Turtle.Move(100)
Turtle.TurnRight()
```

Als je dit programma uitvoert, kun je zien hoe met de schildpad het vierkant één lijn tegelijkertijd wordt getekend. Het resultaat ziet er uit als in de afbeelding hieronder.



Afbeelding 39 – Schildpad tekent een vierkant

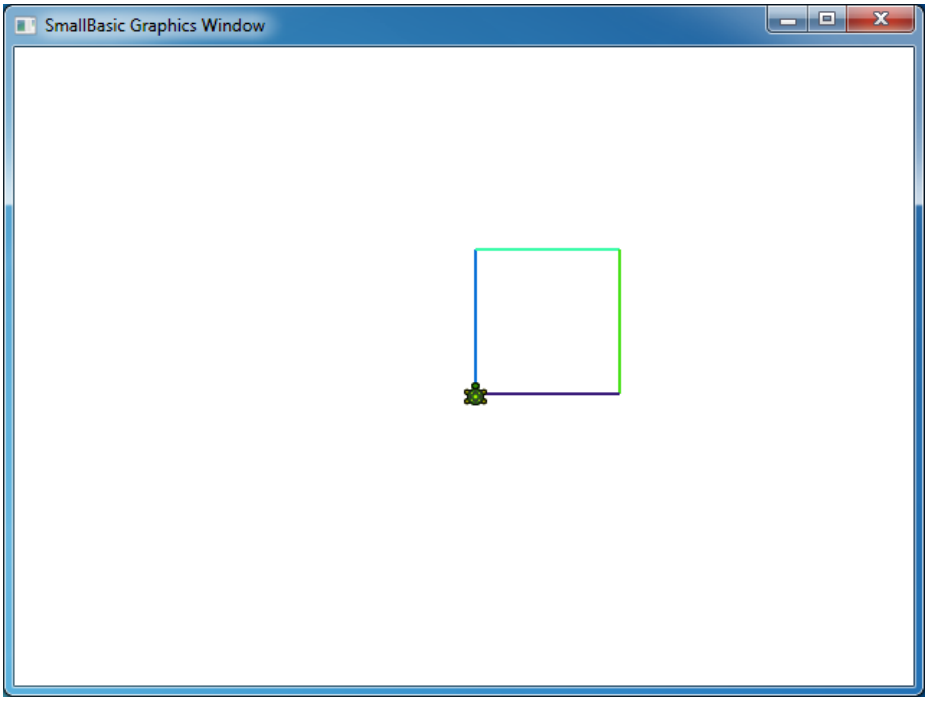
Het is interessant om op te merken dat we dezelfde twee instructies herhaaldelijk uitvoeren, vier keer om precies te zijn. En we weten al dat zulke herhalende opmerkingen kunnen worden uitgevoerd met lussen. Als we daarom het programma hierboven wijzigen om het de **For..EndFor**-lus te laten gebruiken, wordt het programma veel eenvoudiger.

```
 For i = 1 To 4
  Turtle.Move(100)
  Turtle.TurnRight()
EndFor
```

Kleuren wijzigen

De schildpad tekent op hetzelfde GraphicsWindow dat we in het vorige hoofdstuk hebben gezien. Dit betekent dat alle bewerkingen die we hebben geleerd in het vorige hoofdstuk hier nog steeds van toepassing zijn. Het volgende programma tekent bijvoorbeeld een vierkant met elke zijkant in een andere kleur.

```
For i = 1 To 4
  GraphicsWindow.PenColor = GraphicsWindow.GetRandomColor()
  Turtle.Move(100)
  Turtle.TurnRight()
EndFor
```



Afbeelding 40 – Kleuren wijzigen

Meer gecompliceerde vormen tekenen

De Turtle, heeft in aanvulling op de bewerkingen **TurnRight** en **TurnLeft** ook een **Turn**-bewerking. De hoek van de rotatie wordt met één invoer bepaald. Met deze bewerking kun je veelhoeken met een willekeurig aantal zijkanten tekenen. Het volgende programma tekent een hexagoon (een zeszijdige veelhoek).

```
For i = 1 To 6
  Turtle.Move(100)
  Turtle.Turn(60)
EndFor
```

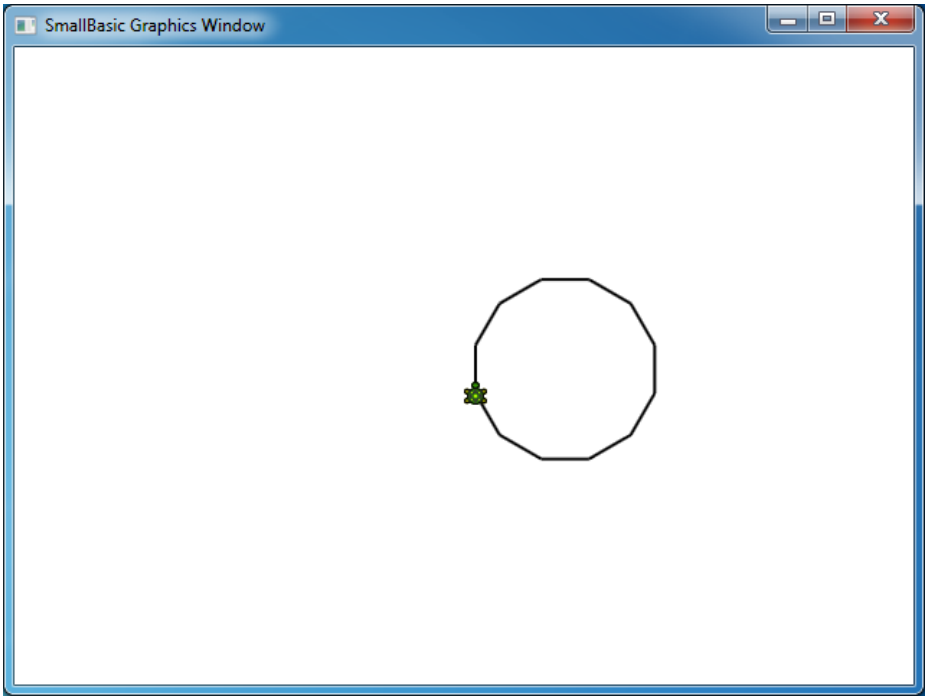
Probeer dit programma zelf om te zien dat er echt een hexagoon wordt getekend. De hoek tussen de kanten is 60 graden en daarom gebruiken we **Turn(60)**. Voor een veelhoek waarvan de zijkanten gelijk zijn, kunnen we de hoek tussen de zijkanten gemakkelijk verkrijgen door 360 te delen door het aantal zijkanten. Met deze informatie en met variabelen kunnen we een vrij algemeen programma schrijven waarmee we een veelhoek met een willekeurig aantal zijkanten kunnen tekenen.

```
zijkanten = 12

lengte = 400 / zijkanten
hoek = 360 / zijkanten

For i = 1 To zijkanten
  Turtle.Move(lengte)
  Turtle.Turn(hoek)
EndFor
```

Met dit programma kun je een veelhoek tekenen door de variabele **zijkanten** te wijzigen. Als je hier 4 opgeeft, krijg je het vierkant waar we mee begonnen. Als je de waarde groot genoeg maakt, zeg 50, is het resultaat niet meer te onderscheiden van een cirkel.



Afbeelding 41 – Een veelhoek met 12 zijkanten tekenen

Met de techniek die we zojuist hebben geleerd, kunnen we de schildpad meerdere cirkels laten tekenen met telkens een kleine verschuiving en dit resulteert in een interessante uitvoer.

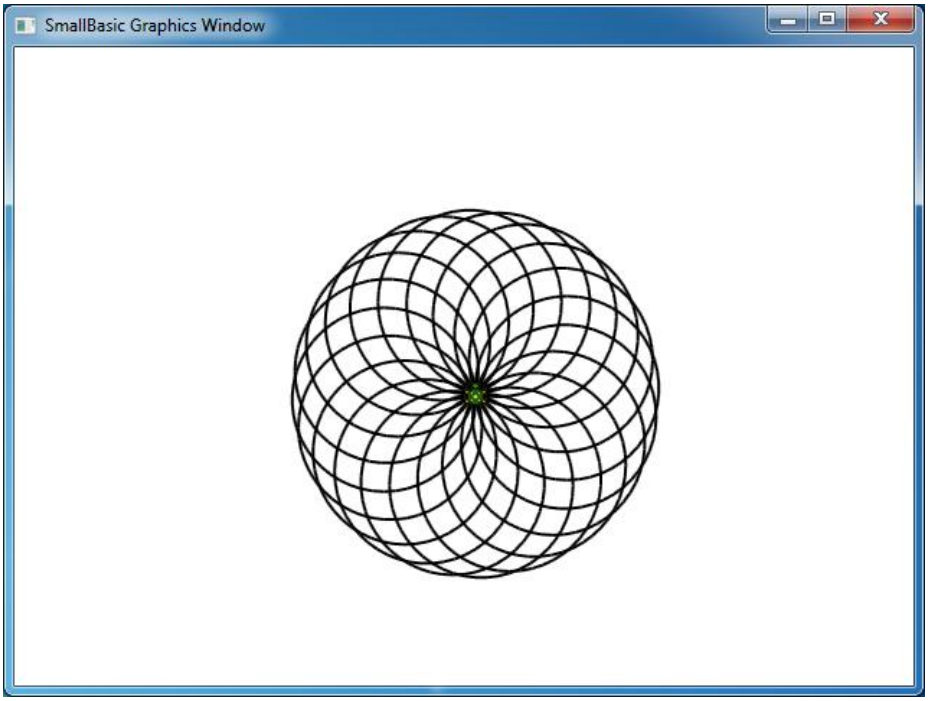
```
zijkanten = 50
lengte = 400 / zijkanten
hoek = 360 / zijkanten

Turtle.Speed = 9
```

```
For j = 1 To 20
  For i = 1 To zijkanten
    Turtle.Move(lengte)
    Turtle.Turn(hoek)
  EndFor
  Turtle.Turn(18)
EndFor
```

Het programma hierboven heeft twee **For..EndFor**-lussen, de een in de andere. De binnenste lus (*i = 1 to zijkanten*) is gelijk aan het veelhoekprogramma en hiermee wordt de cirkel getekend. Met de buitenste lus (*j = 1 to 20*) wordt de schildpad telkens een klein beetje gedraaid wanneer een er een nieuwe cirkel wordt getekend. Dit vertelt de schildpad 20 cirkels te tekenen. Tezamen resulteert dit programma in een erg interessant patroon en dat kun je hieronder zien.

In het programma hierboven verplaatsen we de schildpad door de snelheid in te stellen op 9. Je kunt deze eigenschap instellen op elke willekeurige waarde tussen 1 en 10 om de schildpad zo snel te verplaatsen als dat jij dat wilt.



Afbeelding 42 – In rondjes lopen

In alle richtingen verplaatsen

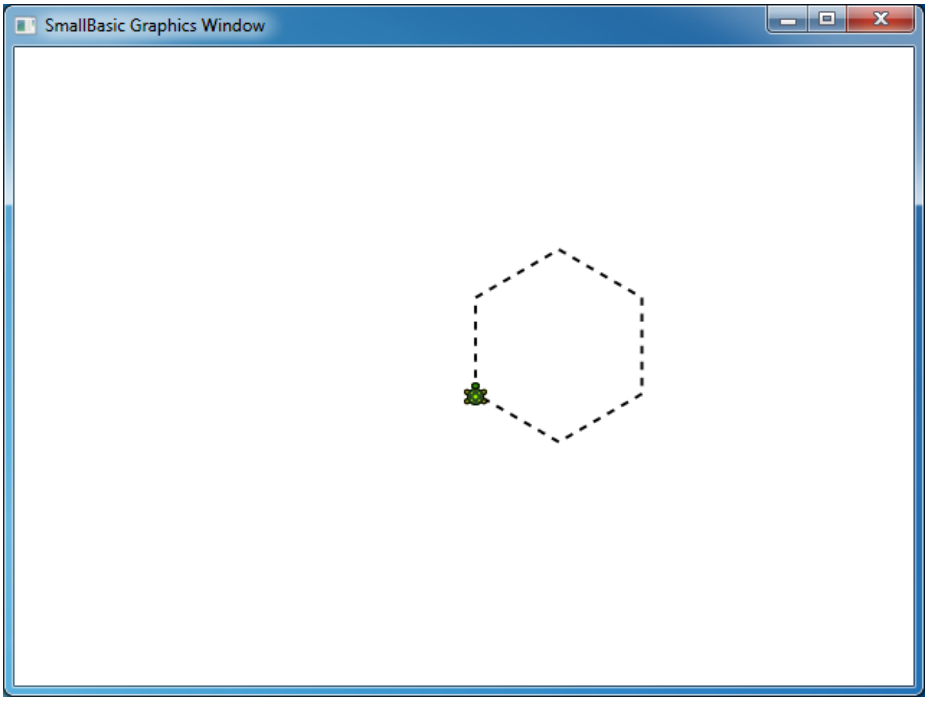
Je kunt de schildpad laten stoppen met tekenen met de bewerking **PenUp**. Hiermee kan de schildpad overal op het scherm worden verplaatst zonder dat er een lijn wordt getekend. Met het oproepen van **PenDown** laat je de schildpad weer tekenen. Dit kun je gebruiken om interessante effecten, zoals stippellijnen, te verkrijgen. Hier volgt een programma waarin dit wordt gebruikt voor het tekenen van een veelhoek met stippellijnen.

```
zijkanten = 6

lengte = 400 / zijkanten
hoek = 360 / zijkanten

For i = 1 To zijkanten
  For j = 1 To 6
    Turtle.Move(lengte / 12)
    Turtle.PenUp()
    Turtle.Move(lengte / 12)
    Turtle.PenDown()
  EndFor
  Turtle.Turn(hoek)
EndFor
```

Ook dit programma heeft twee lussen. Met de binnenste lus tekenen we een enkele stippellijn en met de buitenste lus specificeren we hoeveel lijnen er moeten worden getekend. In ons voorbeeld hebben we 6 gebruikt voor de variabele **zijkanten** en daarom wordt de hexagoon met stippellijn als hieronder getekend.



Afbeelding 43 – PenUp en PenDown gebruiken